Paper 9 - Compiler Construction

Course Description: The course is intended to teach the students the basic techniques that underlie the practice of Compiler Construction. The course will introduce the theory and tools that can be employed in order to perform syntax-directed translation of a high-level programming language into an executable code. These techniques can also be employed in wider areas of application, whenever we need a syntax-directed analysis of symbolic expressions and languages and their translation into a lower-level description. They have multiple applications for man-machine interaction, including verification and program analysis. In addition to the exposition of techniques for compilation, the course will also discuss various aspects of the run-time environment into which the high-level code is translated. This will provide deeper insights into the more advanced semantics aspects of programming languages, such as recursion, dynamic memory allocation, types and their inferences, object orientation, concurrency and multi-threading.

Aims and Objectives: Students successfully completing this course should be able to:

- Understand programming language concepts deeply.
- Understand processing of programming languages by computers.
- Have full command on techniques used by Translator software.

Course Contents: Introduction to Translators, Compiler, Interpreter, Assembler, Context of Compiler, Pre-processor, Assembler, Linker, Loader, Compiler introduction: Analysis-Synthesis Model of Compiler, Phases of Compiler, Two-Pass Assembly, Physical Organization of Compiler, Cousins of compiler, Compiler-Compilers, Lexical Analysis: Role of Lexical Analyzer, Lexical Error Handling, Buffering Issues in Lexical Analyzer, Lexical Analyzer Implementation (Hand coding, Lex), Syntax Analysis: Introduction to Top-Down and Bottom-Up Parsers, Recursive-Descent Parsers, Predictive Parsers, Non-Recursive Predictive Parser, Shift-Reduce Parser, Operator Precedence Parsers, LR Parsers, LL(1) Grammars, LR(1) Grammars, YACC, Syntax Error Handling, Type Systems, Symbol Table Management, Runtime Environment, Intermediate Code: Triples, Indirect Triples, Quadruples, Symbol Table: Techniques such as Lists and Hash Tables, Code Optimization, Code Generation.

Text Books:

1. *Watson, D. (2017). A Practical Approach to Compiler Construction. Springer.*
2. *Mogensen, T. Æ. (2011). Introduction to compiler design. Springer Science & Business Media.*
3. *Dave, P. H., & Dave, H. B. (2012). Compilers: Principles and Practice. Pearson Education India.*
4. *Puntambekar, A. A. (2009). Principles of compiler design. Technical Publications.*
5. *Cooper, K., & Torczon, L. (2011). Engineering a compiler. Elsevier.*